REMARKS

Applicant amended independent claim 1 to clarify that each of the plurality of windows of registers is associated with a corresponding thread, and further clarify that each register in the windows of registers is relatively addressable by the corresponding thread, as well absolutely addressable by two or more of the threads executing on the multithreaded process. Support for this clarification is found, for example, at page 10, lines 15-30, of the filed application. Applicant similarly amended independent claims 12 and 21.

Also, applicant amended independent claim 12 to add the word "circuit" after the term "control logic," as was suggested by the examiner, and amended claims 13, 15 and 19 to make the language recited therein consistent with the language recited in independent claim 12. Further, applicant amended claims 15, 18, and 19 to correct their dependency. Applicant amended independent claim 21 to change the wording "computer readable medium" to "computer readable storage medium." After these amendments, claims 1-8 and 10-21 are pending. Claims 1, 12 and 21 are independent.

The examiner maintained his rejection of claims 1 and 21 under 35 U.S.C. §101 on the ground that the claimed invention is directed to non-statutory subject matter.

Specifically, the examiner states:

> 3. As to claims 1,21, Claims 1, 21 are not limited to tangible embodiments. In view of applicant's disclosure, specification page (2), line (3-15), the method or the medium is not limited to tangible embodiments, instead being defined as including both tangible embodiments (e.g., (hardware based processor)) and intangible embodiments (e.g., (internet). Although the claim recite the instructions cause the access of absolutely address and relatively address, based on broadest interpretation, it is read as intended use , not a positive limitation. The focus is not on the steps or feature taken to achieve the final result which is useful, tangible, and concrete, but rather the final result achieved which is useful, tangible, and concrete (see page 20, 101 Interim Guidelines published at uspto.gov). No final result which is useful, tangible, and concrete can be found in the claims. Therefore, it is directed non-statutory subject matter. (Office Action, page 2)

Applicant's independent claim 1 is performed in a parallel multithreaded processor, which is a tangible device. That parallel multithreaded processor includes a plurality of windows of registers, all of which are tangible, in which registers in those windows are relatively addressable by a corresponding thread, and are also absolutely addressable by two or more

threads. Moreover, the tangible multithreaded processor recited in applicant's claim 1 is one in which the absolutely addressable feature is performed by providing the register's exact address in an instruction.

Accordingly, applicant's claim 1 recites a concrete, useful and tangible action, which is all that is necessary to make a claim statutory. Therefore, applicant's independent claim 1 recites statutory subject matter.

Applicant's independent claim 21 recites a "computer readable storage medium" which is a tangible element. Additionally, the instructions stored on the storage medium are such that they cause a multi-threaded processor to perform operations that, as was explained with respect to independent claim 1, result in a concrete, useful and tangible actions. Applicant's independent claim 21, therefore, recites statutory subject matter.

The examiner rejected claims 1-8 and 10-21 under 35 U.S.C. §102(a) as being anticipated by U.S. Patent No. 5,870,597 to Panwar et al. The examiner also rejected claims 1-6, 8, 10-18, 20 and 21 under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,900,025 to Sollars. Additionally, the examiner rejected claims 7, 19 under 35 U.S.C. §103(a) as being unpatentable over Sollars in view of Panwar.

Applicant's amended independent claim 1 recites "accessing, by a thread executing in the multithreaded processor, a register in a register set organized into a plurality of windows of registers, each of the plurality of windows of registers associated with a corresponding thread, each register in the plurality of windows of registers being relatively addressable by the corresponding thread and absolutely addressable by two or more of the threads executing on the multithreaded processor with absolutely addressable comprises providing an exact address of the register with the exact address specified in an instruction."

As provided below, none of the references cited by the examiner discloses at least the feature of "each register in the plurality of windows of registers being relatively addressable by the corresponding thread and absolutely addressable by two or more of the threads executing on the multithreaded processor."

Panwar discloses a processor that speculatively executes instructions that specify logical addresses, and a processor that converts the logical addresses to physical addresses (Abstract). Particularly, as shown in Panwar's FIG. 3, a register set is divided into windows having 32

registers. Different processes or programs executing on a processor 102 can allocate their own independent window (col. 7, lines 43-47). Panwar further discloses that programs executing on a processor 102 access registers through a typical naming convention, such as r0, r1, r2, ..., r30 (col. 7, lines 51-54), and states:

> **The five-bit register addresses encoded in an instruction word specify the instruction's source registers and the destination register. These register specifiers are logical addresses that index registers within the current register window. (Col. 2, line 66 – col. 3, line 3)**

Programs executing on Panwar's apparatus therefore use relative (logical) addresses when specifying the desired registers that the programs are to access.

Panwar further explains:

> **An individual register within window 302 is physically accessible through register address 304 and current window pointer (CWP) 306. Because the window 302 has 32 registers, the register address 304 will be a 5-bit address. A program, however, would access the registers through a typical naming convention such as r0, r1, r2. . . r29, r30, and r31. In this sense, the current window pointer 306 acts as an offset to address the registers contained in the current window 302. While register file 300 has been shown having 128 registers, and window 302 has been shown as having 32 registers, it will be understood that the size of the register file and register windows is a matter of choice depending upon the needs of a particular application, and as such do not limit the present invention.**
>
> **In SPARC, certain instructions and architectural status registers relate to management of the register windows. As discussed above, a current window pointer (CWP) is maintained in a CWP register to track the current location of the window within the register file. A "SAVE" instruction allocates a new register window to the routine executing it, and saves the prior register window by incrementing the CWP register. A "RESTORE" instruction restores the previous register window (i.e., the register window saved by the last SAVE instruction executed by the current process) by decrementing the CWP register. (Panwar's col. 7, line 48, to col. 8, line 5)**

Thus, the CWP holds the current location of the register window used by the currently executing routine or program, and that location is used as an offset value for computing the physical addresses of the registers being accessed. A "SAVE" instruction causes the offset value in the CWP to change by incrementing the current value held in the CWP. A RESTORE instruction causes the offset value in the CWP register to be decremented. At no point does Panwar describe any instruction that specifies a register's absolute address.

Furthermore, Panwar also does not describe that a given register is addressable by two or more threads. Indeed, Panwar indicates that a motivation for defining register windows is to partition the aggregate of available registers into some pre-determined number of contiguous registers so that "[a]t any one time, the register window permits program access to a subset of the total number of registers in the register file" (col. 2, lines 46-48) A routine using one register window would have no need to access a register in some other register window.

Panwar, therefore, fails to disclose or suggest at least the feature of "each register in the plurality of windows of registers being relatively addressable by the corresponding thread and absolutely addressable by two or more of the threads executing on the multithreaded processor," as required by applicant's independent claim 1.

Sollars describes a processor with a number of control registers logically organized in a hierarchical manner to control the system, context and threads executed by Sollars' processor (see, for example, the Abstract and col. 1, line 60 to col. 2, line 6.) Specifically, Sollars explains that "[p]rimary control register file 20a comprises a plurality of control registers for performing the conventional functions of storing control and status information of executing processes (col. 5, lines 32-35) and describes the register file as follows:

> **FIG. 2 illustrates a logical view of the control registers under the presently preferred embodiment of the present invention. As shown, control registers of primary control register file 20a are organized into registers sets 102, 104, and 106, which in turn are organized into a hierarchy having three control register levels, i.e. a system level, a context level, and a thread level. At the highest level is a set of control registers 102 for controlling overall system operation. At the second highest level are multiple sets of control registers 104 for controlling concurrent execution of processes in multiple peer contexts. At the third level are multiple sets of control registers 106 for controlling concurrent execution of multiple peer process threads of the concurrently executing contexts. (col. 6, lines 35-49)**

Thus, Sollars' control register file is used to control system operation, including the execution of threads. For example, Sollars describes how the control register file can be used to allocate system resources. With reference to allocating resources for executing threads, Sollars explains:

> **As shown in FIG. 12a, similarly, upon receipt of a valid request to create a new thread, i.e. from a thread with the proper context privilege, step 228, the context checks to determine if all thread level control register sets 106**

have been allocated, step 230. If all thread level control register sets 106 have been allocated, the context further determines if the execution priority of the "new" thread requested is higher than at least one of the allocated threads, step 232. If the relative priority determination is unfavorable, the context simply queues the new thread request in a thread request queue of processor 10, step 234. On the other hand, if the relative priority determination is favorable, the context deallocates and queues the lowest priority allocated thread, step 236.

Upon determining that there is at least one free thread level control register set 106 at step 230, or creating one through step 236, the context allocates a free thread level control register set 106 to the "new" thread, step 238. The context then requests the operating system to cause IFU 12 to reprioritize allocation of its internal resources. (col. 12, line 59 to col. 13, line 11)

While the registers of Sollars' register file 20a can be accessed using the registers' virtual or physical addresses (see, for example, col. 9, lines 1-8), at no point does Sollars describe that a register, forming part of a register set allocated to a particular thread, is accessed by some other thread. In other words, Sollars' registers are not addressable by two or more threads. Indeed, because control registers that are assigned to a particular thread are used primarily to maintain control of that thread, there would be no need for another thread to access those control registers. Accordingly, Sollars does not disclose or suggest at least the feature of "each register in the plurality of windows of registers being relatively addressable by the corresponding thread and absolutely addressable by two or more of the threads executing on the multithreaded processor," as required by applicant's independent claim 1.

Because neither Panwar nor Sollars discloses or suggests, alone or in combination, at least the feature of "each register in the plurality of windows of registers being relatively addressable by the corresponding thread and absolutely addressable by two or more of the threads executing on the multithreaded processor," applicant's independent claim 1 is therefore patentable over the cited art.

Claims 2-8 and 10 depend from independent claim 1 and are therefore patentable for at least the same reasons that independent claim 1 is patentable.

Independent claims 12 and 21 recite "each register in the plurality of windows of registers being relatively addressable by the corresponding thread associated with the respective window of registers and absolutely addressable by two or more of the threads executing on the multi-threaded processor," or similar language. For reasons similar to those provided with

respect to independent claim 1, at least this feature is not disclosed by the cited art. Accordingly, independent claims 12 and 21 are patentable over the cited art.

Claims 13-20 depend from independent claim 12 and are therefore patentable for at least the same reasons as independent claim 12.

It is believed that all the rejections and/or objections raised by the examiner have been addressed.

Canceled claims, if any, have been canceled without prejudice or disclaimer.

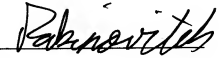
Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

In view of the foregoing remarks, applicant respectfully submits that the application is in condition for allowance and such action is respectfully requested at the examiner's earliest convenience.

Please apply any required charges to deposit account 06-1050, referencing the attorney docket number shown above.

Respectfully submitted,

Date: Nov. 29, 2006



Ido Rabinovitch
Attorney for Intel Corporation
Reg. No. L0080

PTO Customer No. 20985
Fish & Richardson P.C.
Telephone: (617) 542-5070
Facsimile: (617) 542-8906